A full XML-based approach to creating hypermedia learning modules in web-based environments: application to a pathology course

Pascal Staccini[a], MD, PhD, Jean-Charles Dufour[b], MD, Michel Joubert[b], PhD, Jean-François Michiels[c], MD, PhD, Marius Fieschi[b], MD, PhD
[a]STIC, Faculté de Médecine, Université Nice-Sophia Antipolis, Nice, France
[b]LERTIM, Faculté de Médecine, Université de la Méditerranée, Marseille, France
[c]Service d'Anatomie Pathologique, Centre Hospitalier Universitaire, Nice, France

Nowadays, web-based learning services are a key topic in the pedagogical and learning strategies of universities. While organisational and teaching requirements of the learning environment are being evaluated, technical specifications are emerging, enabling educators to build advanced "units of learning". Changes, however, take a long time and cost-effective solutions have to be found to involve our institutions in such actions. In this paper, we present a model of the components of a course. We detail the method followed to implement this model in hypermedia modules with a viewer that can be played on line or from a CD-ROM. The XML technology has been used to implement all the data structures and a client-side architecture has been designed to build a course viewer. Standards of description of content (such as Dublin Core and DocBook) have been integrated into the data structures. This tool has been populated with data from a pathology course and supports other medical contents. The choice of the architecture and the usefulness of the programming tools are discussed. The means of migrating towards a server-side application are presented.

INTRODUCTION

Numerous universities have launched projects targeting the development of hypermedia modules for distance learning or not. One of the essential tasks is to find a suitable system to create, store, combine and deliver these modules[1]. Most of them include a large amount of media objects (such as images, audio, video, animation, simulation…). Basically, the current methods of publishing content on the web are either based on static pages servers or are supported by databases systems. The latter is used by most existing e-learning platforms that offer other functionalities such as control of distributed development, and evaluation and continuous maintenance of the modules[2]. As medical curricula need to be frequently updated, growing attention is given to the scalability of the modules and the interoperability of their content. Furthermore, limited financial and personal resources for system development are forcing universities to develop cost-effective yet extensible concepts for establishing a suitable access and management platform[3,4]. Other aspects have also to be taken into account, such as the availability or not of a network connection, for example. With the need for a stand-alone version of the platform, the question arises concerning the adaptation of the content and the course viewer, that must be as user-friendly as possible. Looking at current e-learning platforms, problems of dependency on content can be identified: courses are not platform independent and course structures are tightly coupled to the specific functionality of the system[3]. Knowledge of the system seems to be the precondition for implementing hypermedia learning modules whereas a pedagogical approach must be considered to be the main guide[5]. Among theories of knowledge, constructivist concepts have attracted a great deal of attention and their application to teaching is based on the principle that knowledge is not passively received but is actively built up by the cognizing subject[6]. This approach suggests that learning modules should present a structured and interactive content that stimulates initiative and research[5].

OBJECTIVES

On the basis of these requirements, we developed a system that accommodates both technical and pedagogical aspects. We worked with pathology teachers who were seeking a course viewer that could integrate texts and images in several resolutions, that could be available through the network and on a CD-ROM without modification, and that could act as a starting-point to engage the learner in active participation, interdisciplinary work and reflection.

METHOD

The project was planned in three phases: 1) defining, gathering and reviewing the learning resources; 2) modelling the course content and the interface of the course viewer; 3) implementing and testing the module. UML notation was used to model the global structure of a course and its components. We implemented it in XML schemes[7] that respect two standards: one for indexing documents (DublinCore[8]) and one for structuring content (DocBook[9]). We chose a client-side architecture with a static pages server. The course viewer used built-in XML parser functions of the navigator, activated by scripts loaded within a single HTML page that manages interface.

RESULTS

Modelling the course components

The analysis of the resources collected by the teachers identified five types of components that constitute the learning environment: 1) the main text that details the topic; 2) the images that illustrate the topic; 3) the links to other resources available on Internet, selected and commented by the teacher; 4) a list of MeSH keywords relevant to the topic; 5) and a set of questions allowing the learner to perform self-assessment. We formalize the relationships between all these components and their attributes in a UML data object model (figure 1). According to this model, a course is a set of chapters. The course and the chapters have a title, the value of which depends on the language. The main components of a chapter that need to be integrated can comprise an electronic presentation, a list of references or a text. These materials have a table of contents and a location. Links of interest and images have also a location. They are described by a short comment. Layers can be placed over each image in order to display either a text or a graphic. Legends can be represented as sets of terms, the first letter of which is displayed at a (X,Y) relative position within the image area. As regards the retrieval keywords and the self-assessment tests, a set of topics has been defined for each chapter. For each topic, combinations of keywords are proposed. Topic tests can be either text-based or image-based. Each item of response has an attribute that indicates its status true or false, and a short comment can be displayed to help students to answer. In order to cater for the multilingual dimension, each entity in the model has an attribute connected with the relevant language. Additional characteristics are related to the glossary of terms used in the text and the relationships between images and parts of the text. Original images were compressed into JPEG format in order to produce three resolutions: 1) thumbnail size (width: 125 px – height: 93 px); 2) standard size (width: 384 px – height: 288 px); 3) enlarged size (width: 768 px – height: 576 px).

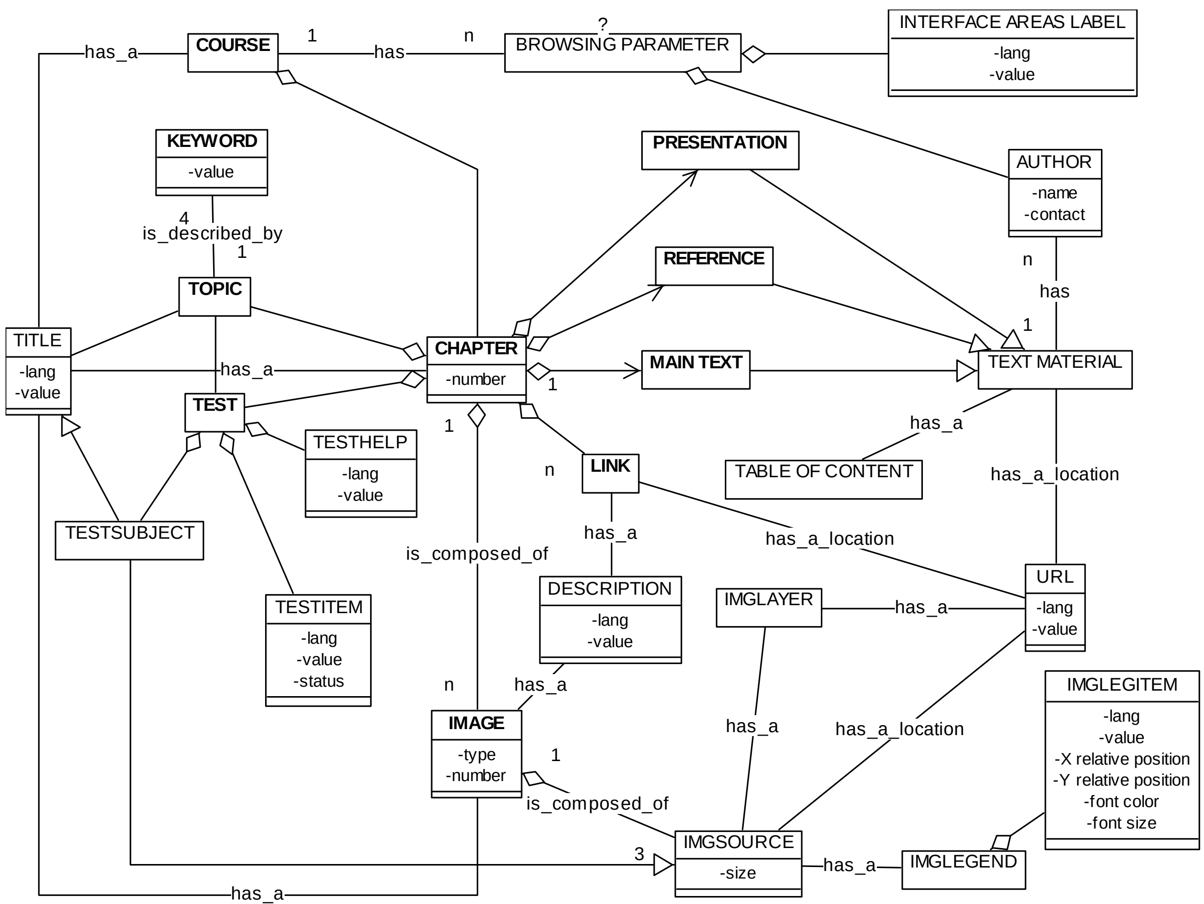

Figure 1: The object data model of the components of a course (UML notation)

The XML implementation

We defined three types of XML files. The first combines the full text, links and keywords. The first set of tags is the set of Dublin Core tags that identifies the author and characteristics of the text. Within the text, DocBook tags have been introduced, so that internal links can be placed in the flow of the text to allow the user to read a definition of a specific term or expression, or view an image (table 1). The last set of tags is related to the links and the keywords selected by the teacher to request the Pubmed database. The second type of XML file stores the list of the courses and the list of the images that compose the module (table 2). The last type of XML files stores the interface parameters and the labels of the different interactive areas.

Table 1: The XML frame that stores the description, the full text and the related components of a course

```
<COURSE>
  <LANGUAGE LANG="FR"/>
  <DUBLINCORE>
    <TOP></TOP>
    <ID></ID>
    <VERSION></VERSION>
    <AUTORITY></AUTORITY>
    <LANGUAGE_COURSE></LANGUAGE_COURSE>
    <DEF></DEF>
    <TYPE></TYPE>
    <PUBLISHER></PUBLISHER>
    <COLLABORATOR></COLLABORATOR>
    <CREATION></CREATION>
  </DUBLINCORE>
  <CHAPTER>
    <TITLE></TITLE>
    <SECT1>
      <TITLE></TITLE>
      <PARA></PARA>
      <SECT2>
        <TITLE></TITLE>
        <PARA>
          <WORD WORDDEF=""></WORD>
          <PICT SRCPICT=""></PICT>
        </PARA>
      </SECT2>
    </SECT1>
  </CHAPTER>
  <MORE>
    <LINK>
      <NAME></NAME>
      <HREF></HREF>
      <COMMENT></COMMENT>
    </LINK>
    <REFERENCES>
       <QUESTION>
         <NUMBER></NUMBER>
         <TITLES></TITLES>
         <MESHTERM></MESHTERM>
         <MESHQUERY></MESHQUERY>
       </QUESTION>
    </REFERENCES>
  </MORE>
</COURSE>
```

Table 2: The XML frames that store the location of the course and image files

```
<LIST>
  <LANGUAGE LANG="FR">
  <COURSE>
    <TITLE></TITLE>
    <LINK></LINK>
    <ISDOC/>
    <ISREF/>
    <ISLNK/>
  </COURSE>
</LIST>

<LIST>
  <LANGUAGE LANG="FR">
  <IMAGE>
    <TITLECOURSE></TITLECOURSE>
    <TITLE></TITLE>
    <LINK></LINK>
  </IMAGE>
</LIST>
```

We worked with the DOM (Document Object Model) to access and manipulate the content of XML files. As defined by the W3C[7], recommendation for a standard DOM covers the ways in which an XML processor (i.e. a browser or an application) exposes data to the outside world. We used the DOM properties and the XML processing features of the IE6 navigator, such as asynchronous loading and parsing of the document, and handling custom HTTP requests for XML documents. XSL was used to specify how the content of the document should be displayed. Various XSL style sheets were defined and linked to the XML files either to populate temporary XML files with data extracted by query (such as the set of images related to a selected chapter) or to present the content of XML documents.

Implementation of the navigation process

Navigation features were implemented using a single HTML file downloaded from a server with JavaScript embedded functions (figure 2). Four navigation areas were defined (figure 3): 1) the “select” area pilots the downloading of XML files and initializes the parameters to provide the user with two basic routes of access to the content of a chapter, either by the type of resource or by the title of chapter; 2) the “topic” area displays, according to the browsing option chosen by the user, either the chapter titles of each type of resource or the resources available for a chapter; 3) the “table of contents” area displays the table of contents of the selected resource, namely, the list of the thumbnails with dynamic filtering capabilities according to an attribute of the image, or the table of contents of the text document with direct linkage to a part of the document; 4) the “content area” contains either the full text, or the image, or the list of links, or the list of keywords.

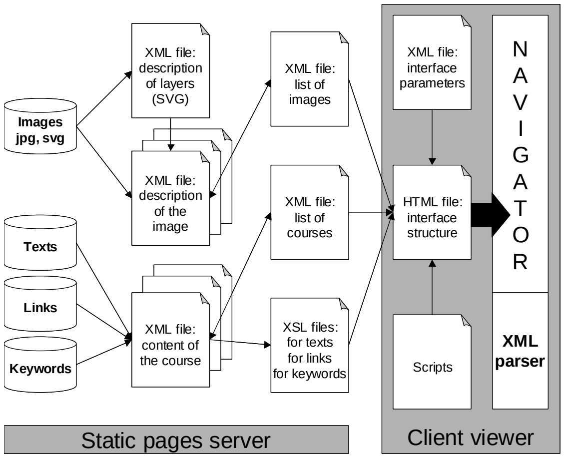

Figure 2: Architecture of the system with types of content, types of XML data files and components of the viewer (parameters, HTML file and scripts)

Some features were implemented to make user navigation more attractive and more relevant. First, a “search” function retrieves all the XML files containing a specific term or expression[10,11]. For each text-based resource, the user can display or mask the table of contents which is dynamically built according to the reading of DocBook tags. Filtering features were implemented, so that images of a course can be displayed according to the criteria of the type of image (radiology, microscopy…). For each image annotated by the author, the viewer enables the user to see or mask the legend and/or the graphic layer that delimits regions of interest. This feature is based on SVG technology (Scalable Vector Graphics) that describes 2D graphics as XML data structures. Cross-navigation between resources was implemented: the author can specify links so that the user can load relevant images while reading the text.

DISCUSSION

The design of web-based learning modules must take into account the learner’s total “virtual” learning environment[10]. The tool we implemented is based on a methodology that guides the teachers in the structuring of their resources according to constructivist learning principles that emphasize intentional learning[5,6]. This method provides students with relevant, selected and classified learning materials, combining traditional text-based documents, annotated images, and appropriate keywords or links to find other resources available on Internet. For teachers, their course preparation is not basically changed. The addition of a new resource involves selecting and describing the new material, storing it in an appropriate directory on the server and encoding the values (description and location) in the XML files. This approach is more flexible than manipulating HTML files. It distinguishes structure of a course from its components and content, allowing easier prototyping and more efficient integration[10,11]. XML technology has been fully employed to structure and store either content or paths to access specific resources. The architecture of the system has been designed according to a client-side approach for implementation either as a stand-alone or as a web-based application without any code change (figure 2). Moreover, the navigator with built-in XML parser functions can be replaced with Java XML parsing technology without restructuring the architecture of the data and the organisation of the XML files. The system may be upgraded using a server-side approach. Objects in the database can be merged for exportation as XML data structures and then be played by the viewer. As regards learning content modelling, our structuring content framework does not address the emerging specifications and the concepts of “units of learning” as described in EML (Educational Modelling Language)[13]. Compliance with these standards of distance learning[14] will be incorporated into the ongoing changes to the system. Nevertheless, we have achieved our target of creating a full XML-based learning environment that can be used in real conditions while remaining generic enough to be adapted to various medical contents. Some limitations still remain. Despite the fact that we have implemented dynamic structures of content, the navigation process remains fixed. The way the user navigates through resources is not traced and the system does not store student profile in order to provide him with adaptive learning features[5,12]. From a pedagogical point of view, this tool can be considered as the first step to prepare students and teachers to work, to learn and to teach within virtual collaborative environments. This XML based system, adapted for pathology, is available on line: http://anapath.unice.fr/test (figure 3). We are currently working to extend the use of this tool to the organization of materials in parasitology and radiology. A specific survey will be performed to query students about the look and feel of the user interface, the relevance of the navigation options, and the level of accessibility of the resources.

References

1. Eskola EL. University students’ information seeking behaviour in a changing learning environment. http:// www.shef.ac.uk/~is/ publications/infres/isic/eeskola.html (last visited: 03/01/2003).
2. Zaina LAM, Bressan G, Silveira RM, Stiubiener I, Ruggiero WV. Analysis and comparison of

distance education environments. International Conference on Engineering Education, Oslo, August 6-10, 2001; 7E8: 19-24

3. Weilt F, Wiesner A, Kelp E. XML based content and its integration into existing e-learning solutions: results and consequences of a Survey. http://daisy.fmi.uni-passau.de/projekte/wwr/ WWRSurveyResultsProceedingsfile.pdf (last visited: 01/03/2003)
4. Fle3>future learning environment http://fle3. uiah.fi/index.html (last visited: 03/01/2003)
5. Nedjl W, Lipeck U. Adaptive hyperbooks: adaptation for project-based learning. http://www.kbs.uni-hannover.de/~henze/diss.pdf (last visited 03/01/2003)
6. Honebein P. Seven goals for the design of constructivist learning environments. In: B. Wilson (ed.), Constructivist learning environments: Case studies in instructional design. New Jersey: Educational Technology Publications, 1996; 11-24
7. Bray T, Paoli J, Sperberg-McQueen M. Extensible Markup Language (XML) 1.0 (second edition). W3C Recommendation 6 October 2000 http://www.w3.org/ TR/2000/REC-xml-20001006 (last visited: 03/01/2003)
8. Dublin Core qualifiers. http://dublincore.org/documents/dcmes-qualifiers/ (last visited: 03/01/2003)
9. DocBook. http://www.docbook.org (last visited 1/03/2003)
10. Bearman M, Kidd M, Cesnik B. Methodology for developing educational hypermedia systems. In: Cesnik B, McCray AT, Scherrer JR, eds. Medinfo'98. Amsterdam: IOS Press, 1998; pp. 720-6
11. Nunes JM, Fowell SP. Developing educational hypermedia applications: a methodological approach. http://www.shef.ac.uk/~is/publications/ infres/paper15.html (last visited: 03/01/2003)
12. Pagesy R, Soula G, Fieschi M. DI2ADEM: an adaptive hypermedia designed to improve access to relevant medical information. Proc AMIA Symp 2000;:635-9
13. Koper R. Educational Modelling Language: adding instructional design to existing specifications. http://www.rz.uni-frankfurt.de/ neue_medien/standardisierung/koper_text.pdf (last visited: 03/01/2003)
14. Friesen N, McGreal R. International Review of Research in Open and Distance Learning. http://www.irrodl.org/content/v3.2/tech11.html (last visited: 03/01/2003)

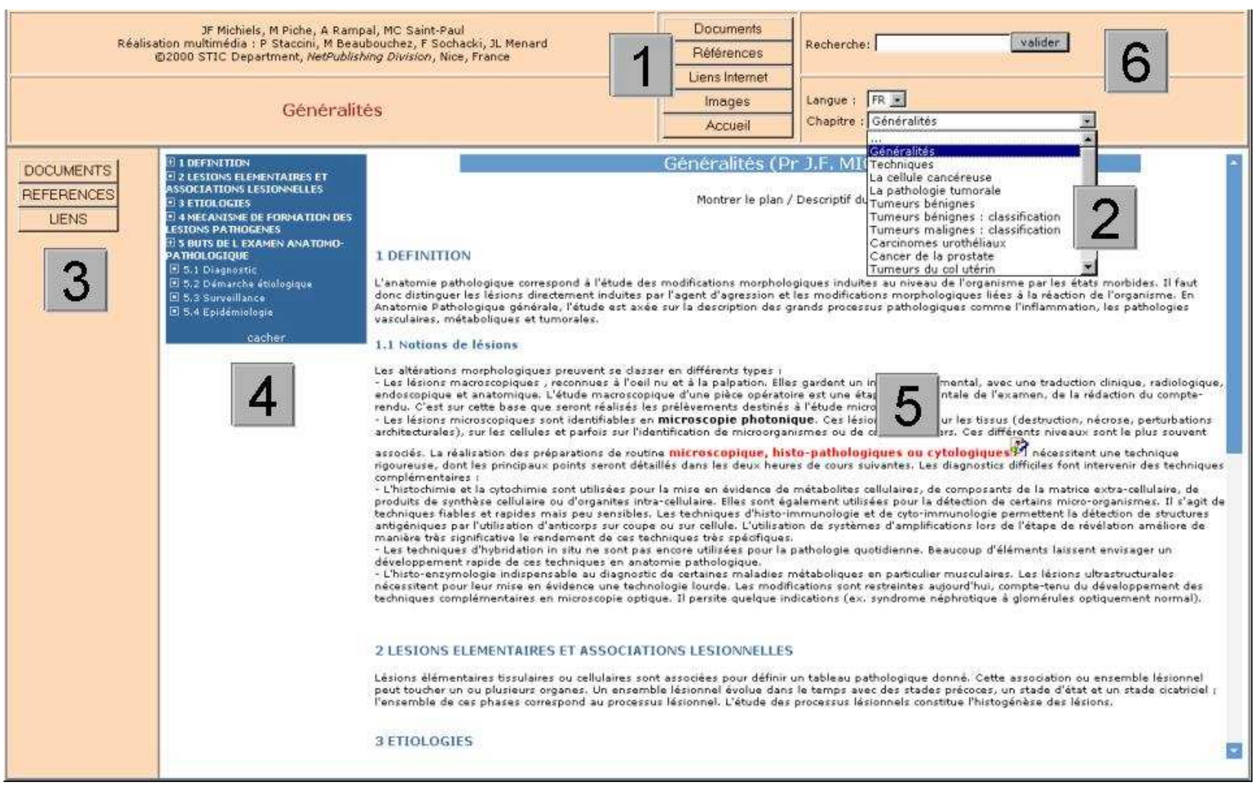

Figure 3: Screen-shot of the user interface with: 1) access to resources according to their type; 2) selection of a specific chapter; 3) dynamic list of the available resources of a selected chapter; 4) dynamic table of contents of the text; 5) links to relevant images or definition of terms; 6) search feature accessing all XML files